Developer Note

AppleVision 1710AV and 1710 Displays

Developer Press

Apple Computer, Inc.
20525 Mariani Avenue
Cupertino, CA 95014
408-996-1010

Simultaneously published in the United States and Canada.

Contents

Figures and Tables

About This Note

The *AppleVision 1710AV and 1710 Displays Developer Note* describes design features of the AppleVision 1710AV Display, a new Apple audio/video display, and of the AppleVision 1710, a similar display with video capabilities only.

This note assumes that you are familiar with the functionality of and programming requirements for Apple Macintosh computers.

The note consists of three chapters, a glossary, and an index.

- Chapter 1, “Overview of the AppleVision 1710AV and 1710 Displays,” gives a hardware overview which includes information about the display’s operating modes; gives a software overview; discusses CPU, video card, and operating system compatibility; provides connection specifications for the displays; and concludes with tips for developers.
- Chapter 2, “Hardware Interface,” describes the hardware interface for the video port, the ADB ports, and the sound ports.
- Chapter 3, “Application Program Interface,” discusses those architectural components that developers can currently access.

Conventions Used in This Note

The following conventions are used throughout this note.

Note
This type of note contains information of general interest. ◆

IMPORTANT
A note like this contains important information that you should read before proceeding. ▲

▲ **W A R N I N G**
A warning like this directs your attention to something that could damage software or hardware or that could result in loss of data. ▲

Terms shown in **boldface** type in the first reference are terms defined in the glossary.

A special font, `Courier`, is used for characters that you type, or for lines of program code. `It looks like this`.

List of Abbreviations

This developer note contains the following abbreviations.

CIE	Commission International d'Eclairage (International Commission on Illumination)
CRT	cathode ray tube
dB	decibel
DDC	display data channel
HDTV	high-density television
ICC	International Color Consortium
LED	light emitting diode
MPCD	mean perceptible color difference
OSD	on-screen display
RAM	random access memory
RMS	root mean square
SPL	sound pressure level
VESA	Video Electronics Standards Association
VGA	video graphics adapter
VPT	Virtual Photometry Technology
VRAM	video RAM

Other Reference Material

Apple Developer Press publishes a variety of books and technical notes designed to help third-party developers to design hardware and software products that are compatible with Apple computers. Readers should be familiar with the following documents that can be found in electronic form in the latest Reference Library edition of the Developer CD Series:

- *AV Architecture Developer Note.* This note is a companion to the *AppleVision 1710 and 1710AV Displays Developer Note*, and you should refer to both books for a thorough understanding of the AppleVision displays. The note gives an overview of AV Architecture concepts and AV applications software. It provides detailed information about the architecture's Panel, Engine, Port, and Device Components, and gives an overview of Device Manager Component features as they relate to the AV Architecture.
- *Display Device Driver Guide, Device Support for the Display Manager Developer Note* describes device support for the Macintosh Display Manager.

- *Designing PCI Cards and Drivers for Power Macintosh Computers*, # R0650LL/A, describes the Macintosh implementation of the Peripheral Component Interconnect (PCI) local bus. Chapter 11 of this book is of particular interest to developers working with AppleVision displays and the AV Architecture.

Inside Macintosh is a collection of books, organized by topic, that describe the system software of Macintosh computers. Readers should also be familiar with the following publications that can be found in the *Inside Macintosh* CD.

- *Inside Macintosh: QuickTime*
- *Inside Macintosh: More Macintosh Toolbox*
- *Inside Macintosh: Devices* documents the last version of the Device Manager before it was enhanced to support PowerPC native drivers.
- *Inside Macintosh: Advanced Color Imaging*

You should also refer to the third edition of *Designing Cards and Drivers for the Macintosh Family*, published by the Addison-Wesley Publishing Company, Inc.

Most of the publications listed are available from APDA. Refer to the next section for the APDA address and phone numbers.

For More Information

APDA offers convenient worldwide access to hundreds of Apple and third-party development tools, resources, and information for anyone interested in developing applications on Apple platforms.

To order products or to request a complimentary copy of the *APDA Tools Catalog*, contact

APDA
Apple Computer, Inc.
P.O. Box 319
Buffalo, NY 14207-0319

Telephone	1-800-282-2732 (United States) 1-800-637-0029 (Canada) 716-871-6555 (International)
Fax	716-871-6511
AppleLink	APDA
America Online	APDAorder
CompuServe	76666,2405
Internet	APDA@applelink.apple.com

For the latest specifications and information about the Display Data Channel (DDC) contact

VESA
2150 North First Street, Suite #360,
San Jose, CA 95131-2020

Telephone	408-435-0333
Fax	408-435-8225

CHAPTER 1

Overview of the AppleVision 1710AV and 1710 Displays

The AppleVision 1710AV and 1710 Displays are multiple-resolution color displays with extensive software control features. They extend the traditional display concept and provide customers with an affordable step up to a high-quality multimedia input/output (I/O) device.

The AppleVision 1710AV Display has both audio and video capabilities. It reemphasizes Apple's commitment to the integration of sound input and sound output with a desktop computer, and it is designed to take advantage of today's multimedia applications, such as PlainTalk voice control, text-to-speech technology, and hands-free telephony, as well as emerging **MovieTalk** technologies for video conferencing. It has a built-in PlainTalk-compatible microphone, integrated stereo speakers, and enhanced audio and I/O capabilities.

The AppleVision 1710 Display shares all the video capabilities of the AppleVision 1710AV. However, it does not have audio capabilities.

Chapter 1 of this note provides an overview of the AppleVision 1710AV and 1710 Displays. It includes the following information:

- a general description of the AppleVision display hardware, including I/O connectors, controls, display specifications, and power requirements
- an overview of the display software elements
- information about compatible computers, video cards, and operating systems
- a discussion of hardware and software opportunities available to developers
- display connection specifications, including an overview of design techniques used to develop cards and drivers for the new displays, to design smart displays, and to design displays with Display Data Channel (DDC) capabilities that allow them to function with PC-compatible computers

Note
The information in this chapter applies to both the AppleVision 1710AV Display and the 1710 Display, unless otherwise indicated. ◆

Hardware Overview

The AppleVision 1710AV and 1710 Displays are new 17-inch RGB (red, green, blue) multiple-resolution displays. Both displays have the following features:

- EPA **Energy Star**™ compliance
- a standard Macintosh power input cable
- two ADB (Apple Desktop Bus) ports to connect ADB devices such as keyboard and mouse
- easily accessible controls to change video brightness, contrast, color depth, and resolution
- front panel controls for the on-screen display (OSD)

The AppleVision 1710AV has the following additional features:

- an output port for headphones or external speakers
- an input port for an external microphone or other sound source
- a built-in microphone, with PlainTalk capability and a light-emitting diode (LED) indicator
- a built-in stereo speaker system
- easily accessible controls to adjust sound volume, to mute sound, to adjust bass and treble, and to enable or disable the microphone

Both displays feature true multiple-scan capability. Table 1-1 lists some of the resolutions supported.

IMPORTANT
The AppleVision displays are not limited to the timing modes listed in Table 1-1. The displays will synchronize over timing ranges 50-120 Hz for vertical refresh, and 28-82 kHz for horizontal scan. ▲

Table 1-1 Supported screen resolutions

Mode	Pixel resolution	Vertical refresh rate	Horizontal scan rate
VGA	640 × 480	60 Hz	31.5 kHz
Macintosh	640 × 480	66.67 Hz	34.97 kHz
VESA	800 × 600	60.31 Hz	37.9 kHz
VESA	800 × 600	75 Hz	46.9 kHz
Macintosh	832 × 624	74.55 Hz	49.7 kHz
Macintosh	1024 × 768	74.93 Hz	60.24 kHz
VESA	1024 × 768	60 Hz	48.4 kHz
VESA	1280 × 1024	60 Hz	64.3 kHz
VESA	1280 × 1024	75.03 Hz	79.98 kHz

The displays are ergonomically designed with an integral tilt-and-swivel base. They interface by means of a video cable with existing Macintosh computers and with PC-compatible computers. Chapter 2, “Hardware Interface,” provides information about interfacing with the displays. Figure 1-1 and Figure 1-2 show front and rear views of the AppleVision 1710AV Display. Figure 1-3 and Figure 1-4 show front and rear views of the AppleVision 1710 Display.

Figure 1-1 Front view of the AppleVision 1710AV Display

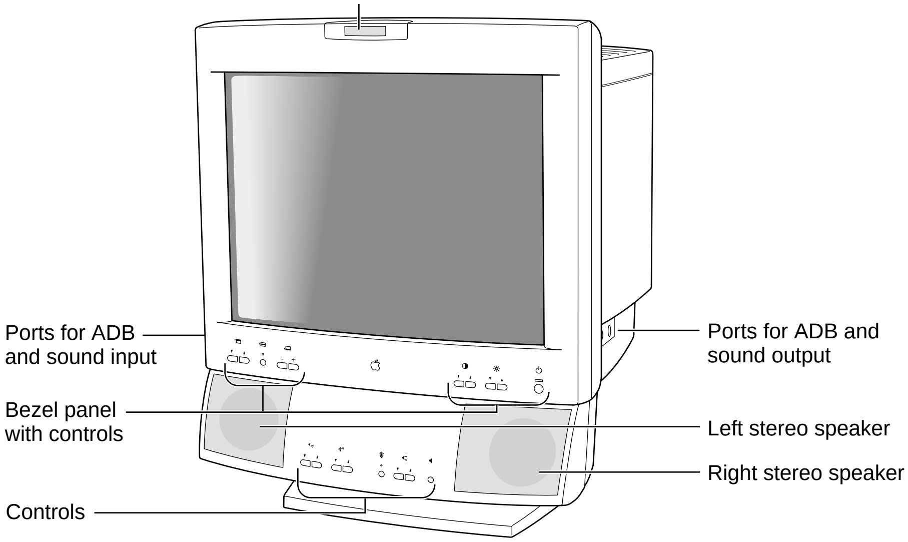


Figure 1-2 Right side and rear panel of the AppleVision 1710AV Display

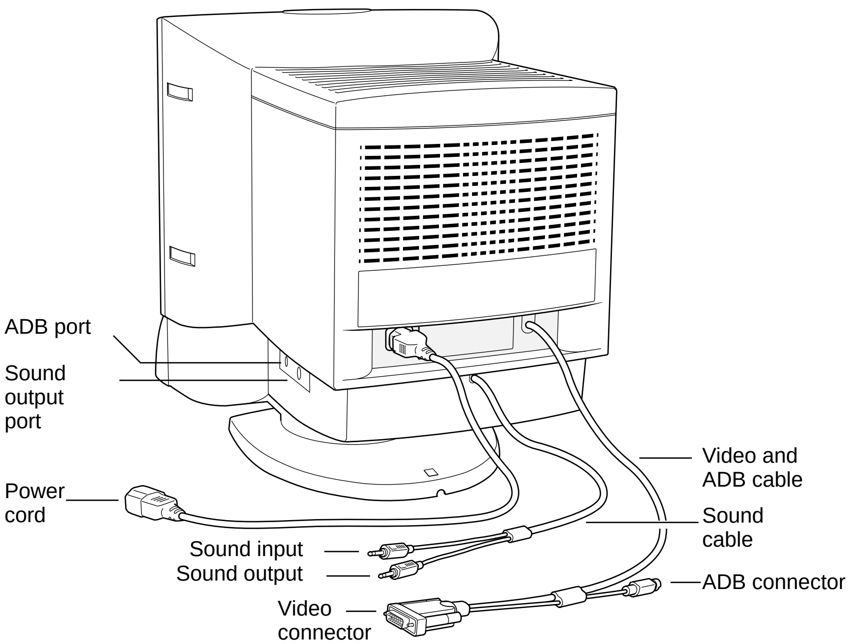


Figure 1-3 Front view of the AppleVision 1710 Display

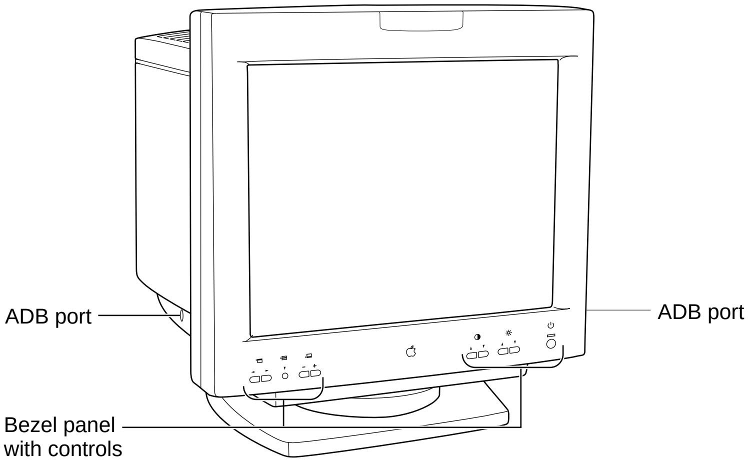
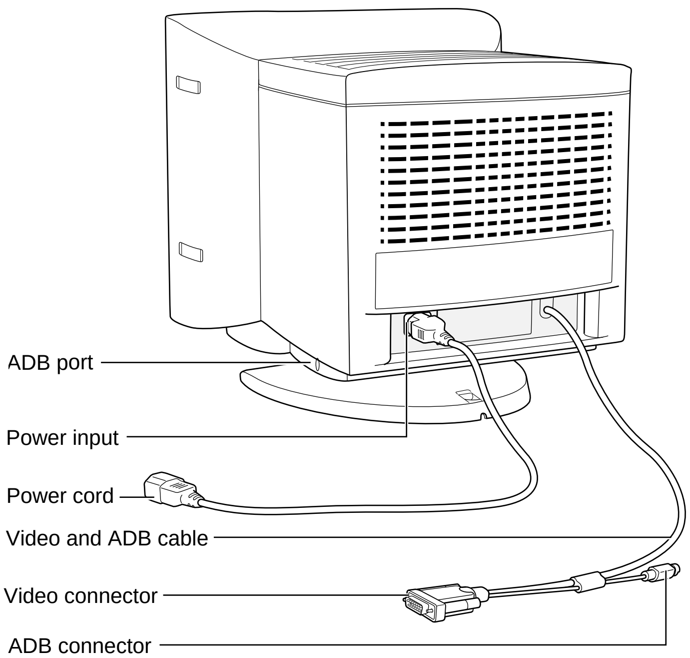


Figure 1-4 Right side and rear panel of the AppleVision 1710 Display



Operating Modes

AppleVision displays have a feature that allows them to work in two different modes, depending on whether or not the AppleVision software is present. These modes are Remote and Local.

When the display is first powered up, it is in Local mode. If it is connected to a PC-compatible computer, it remains in Local mode. If it is connected to a Macintosh computer, but the Macintosh AppleVision software is not installed, it also remains in Local mode. If the display is connected to a Macintosh computer with the AppleVision software installed, the software puts the display into Remote mode by sending a series of ADB messages to the display. The display then remains in Remote mode.

In Local mode, the computer does not control the display. When the user presses control buttons on the front of the display, the display handles these actions and makes the required changes to its settings. In addition, when one of the video front panel buttons is pressed, the on-screen display (OSD) appears to provide visual feedback on the changes being made.

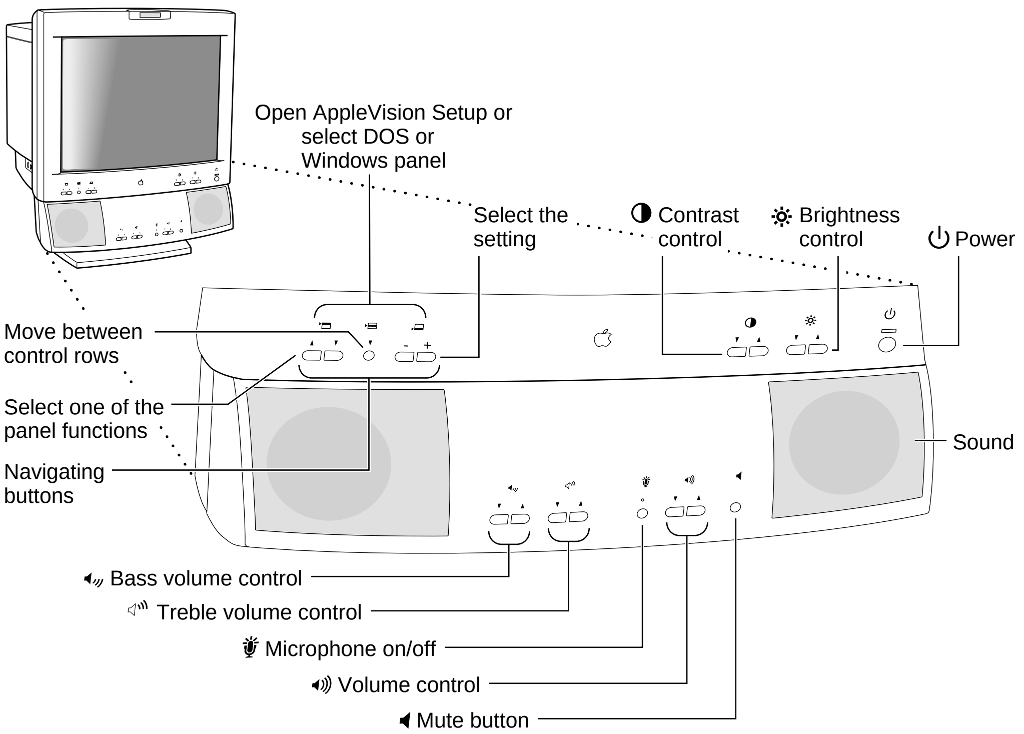
Remote mode allows the AppleVision Setup application to control the settings of the display. In addition, the OSD does not appear on the screen. Instead, when any front panel buttons are pressed, the information about which button was pressed is sent to the Macintosh over the ADB cable. The Macintosh then commands the display to change its settings appropriately. If the AppleVision Setup application is open when the buttons on the display are pressed, the application reflects the changes caused by the buttons as those changes are made.

Front Panel Controls

The AppleVision 1710AV Display controls are located on the front bezel of the display, as shown in Figure 1-1 on page 4. The AppleVision 1710 Display controls are shown in Figure 1-3 on page 5. Figure 1-5 shows details of these controls. The AppleVision 1710 Display does not have audio capabilities, so the audio controls shown on the lower panel (Figure 1-5) are not present.

The buttons on the front panels of the displays control the various functions electronically through the main processor and related system software, when the displays are in Remote mode. In Local mode, screen brightness and contrast, audio volume, and the microphone are affected directly by these controls. Refer to the previous section, “Operating Modes,” for further information on this subject.

Figure 1-5 Front panel audio and video controls



There are three groups of controls on the AppleVision 1710AV Display's front panel. The AppleVision 1710 Display has two groups of controls.

Both displays have a group of buttons in the upper left corner of the panel. These buttons perform different functions, depending on whether the display is operating with a Macintosh computer with AppleVision software, or with a PC-compatible computer.

- When you are using the display with a Macintosh computer with AppleVision software installed, pressing any one of the five buttons in the upper left corner of the panel launches the AppleVision Setup control panel if it is present in the Control Panels folder.
- When you are using the display with a PC-compatible computer, the buttons enable you to select DOS and Windows panels. If you press any one of the buttons, it opens the on-screen control panels. From these panels you can change the display's setup.
  - The first two buttons select among the three main menus: Geometry, Tools, and Convergence. Geometry is the default that opens up first.
  - The third (center) button allows you to select among the different topics under each menu. For example, in the Geometry menu, you can choose Width, Height, Pincushion, Vertical Shift, Horizontal Shift, or Rotate. In the Tools menu, you can chose Color Temperature, Overscan, or Scan Rate. (You can only display the scan rate when you select Scan Rate. You cannot adjust the scan rate, as the display adjusts to the scan rate coming from the video card.) In the Convergence menu,

you can select Vertical and Horizontal. The panels displayed when you select the different panels do not look the same as the panels used for the Macintosh computer.

- The fourth and fifth buttons allow you to select the value for the different topics, so that you can adjust Width, Height, and so forth. If you press the + button, the value increases, and if you press the – button, the value decreases.

Both displays have a group of buttons on the upper right corner of the panel. These buttons are used in both Macintosh and PC configurations.

- Contrast control. There are two contrast control buttons. The one with the up arrow increases contrast. The one with the down arrow decreases contrast. The control process is continuous.
- Brightness control. There are two brightness control buttons. The one with the up arrow increases brightness. The one with the down arrow decreases brightness. The control process is continuous.
- Power on and off. This button turns power to the display on and off. Power to the display comes on only when the video signal is present.

The buttons on the lower panel are present only on AppleVision 1710AV Displays. They are used in both Macintosh and PC configurations to control the audio functions of the display.

- Bass control. These buttons adjust bass. The one with the up arrow increases bass. The one with the down arrow decreases bass.
- Treble control. These buttons adjust treble. The one with the up arrow increases treble. The one with the down arrow decreases treble.
- Microphone on and off. This button turns the internal microphone on and off. An amber light above the button comes on when the microphone is on.
- Volume. These buttons increase or decrease the overall volume of the sound output. The one with the up arrow increases volume. The one with the down arrow decreases volume.
- Mute button. The first time you press it, sound is muted, the second time, sound is turned on again. You can also turn sound on again by pressing the increase volume button.

For detailed information about the controls, you should refer to the User's Guides for the AppleVision 1710AV Display and the AppleVision 1710 Display.

I/O Connections

The AppleVision 1710AV (Figure 1-2 on page 4) and 1710 (Figure 1-4 on page 5) Displays have I/O connections on the back panel and also on the side panels.

Both displays have the following connections:

- The video and ADB cable is permanently attached to the rear panel of the display. There are two connectors at the end of this cable. One is a DB-15 video connector. The other is a standard ADB connector. Both connectors plug into the back panel of the Macintosh computer. If you are using the AppleVision display with a PC-compatible computer, you will not use the ADB connector at all. To connect the display to the computer, connect the video connector to a PC adapter, and then plug the PC adapter into the PC video port.

- There is a standard power connector that plugs into the back of the display.

The AppleVision 1710AV Display also has the following sound connection:

- The sound cable is also permanently attached to the rear panel of the display. It is a split cable with two connectors, one for sound input and the other for sound output. You plug these connectors into your computer if you want to use the built-in microphone and speakers on the AppleVision 1710AV Display.

Both displays have the following ADB ports on the side panels:

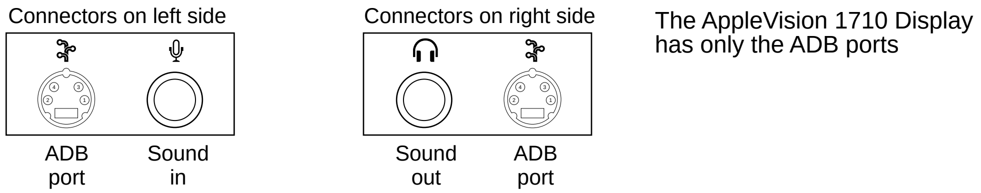
- The ADB ports, on the left and right side panels, enable you to connect a mouse or a keyboard to either side of the display.

The AppleVision 1710AV has the following sound ports on the side panels:

- The sound output port, on the right side panel of the display, is a line-level output that supports headphones and external speakers. Plugging in a sound output device to this port does not automatically disable the built-in speakers. You may select an option from the Sound panel that allows you to disable the speakers when the headphones are plugged in.
- The sound input port, on the left side panel, allows you to connect an external microphone or other line-level sound source, such as a CD or cassette player, to the display.

Figure 1-6 shows the relative positions of the ports on the side panels.

Figure 1-6 AppleVision display I/O ports



Chapter 2, “Hardware Interface,” provides the interface specifications for these connectors.

Speakers and Microphone

The features described in this section are available only with the AppleVision 1710AV Display.

- The integral stereo speakers feature a ported (bass reflex) chamber design. They have a response close to high fidelity, and deliver 92 dB SPL (sound pressure level) at 1 kHz, at 0.5 meters. The frequency response is 70Hz to 20 kHz, ± 6dB.

- The headphone/speakers jack has an output level of 1.75 Vpp into a 4-ohm load, and 2 Vpp into a 10-kohm load.
- The integral microphone is a directional microphone, optimized for use with speech recognition programs. It is particularly effective in noisy office environments.
- The display accepts an audio input signal of up to 4 Vpp (line level, low sensitivity), or as low as 20 mVpp (microphone level, high sensitivity) without clipping the signal.

Specifications

Table 1-2 lists the environmental specifications for the display.

Table 1-2 Environmental specifications

Item	Specification
Temperature	10° to 40° C (50° to 104° F) — operating 0° to 60° C (32° to 140° F) — storage -40˚ to 60˚ C (-4˚ to 140˚ F) — shipping
Relative humidity	10% to 80% noncondensing — operating 5% to 90% noncondensing — storage 5% to 95% noncondensing — shipping
Operating altitude	0 to 10,000 feet (0 to 3048 meters)
Shipping altitude	0 to 35,000 feet (0 to 10,670 meters)

Table 1-3 lists the AC power requirements for the display.

Table 1-3 AC power requirements

Category	Requirement
AC input range	100–200 volts AC, auto select
Input surge voltage	3 kV
Input line transient immunity	RF level of 3 volts/meter, from 26 MHz to 1 GHz
Peak inrush current	40 amps peak, all load and line conditions
Input line frequency	50– 60Hz, single phase
Line drop out immunity	20 milliseconds (minimum), 90 VRMS input, maximum load
Input power under maximum load	130 watts (maximum), all line and load conditions

Table 1-3 AC power requirements (continued)

Category	Requirement
Minimum input/output power efficiency	75%, all line conditions, maximum load
Line voltage transient response	±10% instantaneous variation in average input line voltage, applied for 100 ms, with no visible effects of transient in display
Load regulation	Adequate for proper operation of display-related circuitry under all conditions

Macintosh Software Overview

The software that ships with the AppleVision 1710AV and 1710 Displays provides the software interface between the user and the display when the display is connected to a Macintosh computer system.

The software performs the following functions:

- It provides basic geometry control for a variety of screen functions, such as brightness and contrast, **pincushioning**, **overscan**, and so forth.
- It provides basic sound control for a variety of functions such as volume and bass, and treble levels. This applies only to the AppleVision 1710AV Display.
- It provides support for **Virtual Photometry Technology (VPT)**, with the capability of creating **white point**, **ColorSync profiles**, **ambient light** settings, and compensating for CRT aging.
- It gives quick access to frequently used control functions through the Control Strip.
- It supports energy saving with the **Energy Saver**.

An application interface specific to the displays is part of the software package. This is the AppleVision Setup application program, and it resides in the Control Panels folder.

Compatibility Issues

This section deals with AppleVision compatibility issues, in terms of the CPUs, video cards, and versions of the Macintosh operating system that support the AppleVision displays.

CPU Compatibility

Table 1-4 lists the CPUs that support the AppleVision displays. The CPUs listed have been tested in AppleVision configurations.

Table 1-4 CPU configurations supporting the AppleVision display

Powerbook	Quadra, Centris, Performa	Power Macintosh
280*	605	6100/60
280c*	610	6100/60AV
520	630	6300
520c	630 DOS compatible	7100/66
540	650	7100/66AV
540c	660AV	7100/80
	700	7100/80AV
	800	7200
	840AV	7500
	900	8100/80
	950	8100/80AV
		8100/110
		8100/110AV
		8500
		9500
		Power Macintosh upgrade card

* With Mini Dock and Duo Dock II

Video Card Compatibility

In addition to the on-board video provided by the CPUs listed in Table 1-4, plug-in video cards, such as the NuBus 24AC video card, also support AppleVision displays. For additional information about supporting the AppleVision displays using plug-in video cards, refer to “New Display Connections Specifications” on page 14.

Apple video cards 8.24, 8.24GC, and 4.8 do not support the AppleVision displays. In addition, plug-in video cards that do follow the new guidelines defined in the “Graphics Drivers” section of *Designing PCI Cards and Drivers for Macintosh Computers* do not support the displays.

Operating System Compatibility

You should use System 7.5 or later with the AppleVision display. If you run earlier versions of the operating system, QuickTime delays the loading of the AppleVision INIT. You may have renamed AppleVision INIT with a name that begins with a letter later than Q (QuickTime). This means that the INIT installs after QuickTime, and QuickTime moves the AppleVision components, delaying booting.

Opportunities for Developers

This section discusses the opportunities for developers working with the AppleVision 1710AV and 1710 Displays to develop software and hardware.

Software Opportunities

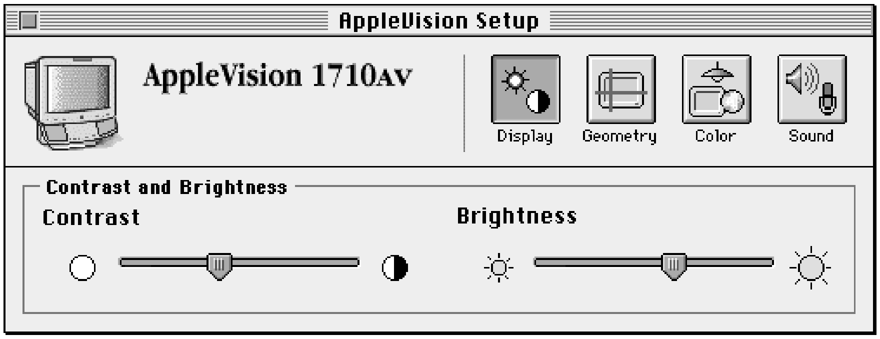
The AppleVision 1710AV and 1710 Displays are the first of a new generation of Apple displays. They allow unprecedented control of the computer display from Macintosh software, and feature excellent integration with the Display Manager and system software. The software shipped with the display is noteworthy because it takes full advantage of Apple's new AV Architecture (described in the *AV Architecture Developer Note*) and it can be used on all CPUs listed in Table 1-4.

If you are developing software, you can access the engine layer of the AppleVision software, find engines for smart displays, and control the features of those displays directly. For example, a game developer might want to switch into overscan mode to take advantage of the full screen area of the display.

The application layer consists of application software that can find and display panels. There are currently two applications capable of finding and displaying AppleVision 1710 panels. They are the Sound & Displays application that ships with all new Power Macintoshes, and the AppleVision Setup application that ships with the AppleVision display.

When you install the AppleVision extension in the Extensions folder, these applications are enabled to show the panels that provide control of the Applevision displays. In addition, the AppleVision extension provides similar control over the AudioVision 14 Display. Figure 1-7 shows the brightness and contrast panel from the AppleVision Setup application.

Figure 1-7 Contrast and brightness panel



Hardware Opportunities

Using the guidelines outlined in "New Display Connections Specifications," third-party graphics device developers can develop plug-in video cards that offer compatibility with this new generation of displays.

New Display Connections Specifications

This section details important new standards for Apple Macintosh display connections, and you should refer to it if you are planning and designing new Macintosh video hardware and software products.

The AppleVision 1710AV and 1710 Displays are the first Apple displays to combine smart-display technology and Multiple Scan technology. Because of innovations in the AppleVision 1710 displays, Apple is making some important revisions to its standards for communication between CPUs and displays. These revisions, outlined in the following sections, are important to third parties who produce displays, video output devices, and video drivers.

Note
In the context of this developer note, a smart display is one that can be controlled by the Macintosh computer and that, in the case of the AppleVision displays, communicates with computer by means of the ADB cable. ◆

Background Information

In the past, graphics drivers sensed the type of display attached to the video card by means of three sense lines on the video cable. These lines were encoded to produce a hardware sense code algorithm. With three sense lines, the number of displays that could be identified was limited to seven, plus the instance where no displays were connected.

All video cards had to know the sense codes of all displays connected. Once the sense code was determined, the graphics driver trimmed its list of available timing modes to those that it calculated were possible. Before System 7.1.2, and the introduction of the Display Manager, all functional sResources not disabled by the PrimaryInit were always shown in the Monitors control panel. There was no way to find out which timings had been trimmed, and no way to tell which timing corresponded to a functional sResource.

When the Display Manager was introduced, Apple added two new video driver status calls: `cscGetConnection` and `cscGetModeTiming`. These calls allow the Monitors control panel to show only those modes that are marked as valid. The Monitors panel does this by means of the `DMCheckDisplayMode` call, which in turn calls `cscGetConnection` and `cscGetModeTiming`. Using this method, video cards can have untrimmed functions sResources that correspond to invalid timing modes and do not show up in the Monitors control panel.

You may view the sense code as returning the default timings supported by the display. With Display Manager 2.0, smart displays, like the AppleVision 1710AV and 1710, can enable and disable timings returned by the video driver. The video driver does not have to know which timings are supported by a smart display, since the display itself makes the finals decision as to whether it can support a particular timing. `DMCheckDisplayMode` gets the video driver's best estimate of the supported timings, by means of the safe and valid bits; calls the smart display with this information; the smart display looks at the timing constant returned by the `cscGetModeTiming` call, and then modifies the valid and safe bits. The video driver is not notified of changes made by the smart display.

Designing Cards and Drivers for a New Generation of Displays

The AppleVision 1710 displays are **multiple-scan displays**. As such, they would normally use one of the three Type 6 Extended Sense Codes. However, these new 17-inch displays support any possible resolution that falls within the timing range of 28-82 kHz horizontal scan and 50-120 Hz vertical refresh. It is thus clear that the current Type 6 Extended Sense Codes are insufficient to cover all the possible variations of resolutions supported by these new multiple-scan displays.

The AppleVision 1710 displays, therefore, rely on a new strategy, in which the Display Manager, rather than the graphics driver, makes timing mode decisions. This strategy is outlined in *Designing PCI Cards and Drivers for Power Macintosh Computers*. Refer to the Chapter 11, "Graphics Drivers," and the section "Display Timing Modes," to see how Apple is revising its strategy for timing modes.

Of particular importance in this new strategy is the requirement that the graphics driver report, as available, all timing modes supported by the current graphics card hardware. This allows the software shipped by the display manufacturer to report independently the timing modes it supports, and allows the Display Manager to provide the point where these two sets of timing modes intersect.

The AppleVision 1710 displays return the RGB 13" Type 6 ("straight 6") sense code as a safe choice when the display is connected to a Macintosh computer that does not have the new Display Manager. This also allows the Macintosh to display stable on-screen video early in the boot process. There is also another reason for using this sense code, as described in "Designing Display Data Channel (DDC) Displays" on page 16.

Because the AppleVision 1710 displays, and other future displays from third parties, will rely on this new strategy, all PCI cards as well as Nubus cards intended to support these multiple-scan displays must follow the guidelines outlined in *Designing PCI Cards and Drivers for Power Macintosh Computers*. These changes in the Display Manager work with PCI drivers on all Power Macintosh computers and with Nubus drivers on all color-capable Macintosh computers running System 7.5.

Designing Smart Displays

If you are designing a smart display, that is a display where the Macintosh computer can control some functions of the display via the ADB cable, serial port, or other communications method, the display should include as one of its functions the capability of independently "**wiggling**" or toggling the value of the sense lines. This is done by means of a call to the display's Port Component wiggle selectors, so that the Display Manager can establish the connection between the Port Component and the gDevice associated with the display. (Refer to the *AV Architecture Developer Note* for more information.) If the display cannot implement wiggling, the Sound & Displays control panel will show two display ports (one for the graphics device, one for the display) rather than one combined port, and it will be up to the developer to determine which gDevice is used by the display, if this information is needed.

Note
In some Apple publications, the term "tagging" is used instead of the term "wiggling." ◆

Designing Display Data Channel (DDC) Displays

The AppleVision 1710 display is the first Apple display that provides compatibility with Windows 95 Plug-and-Play via the Display Data Channel (DDC) standard. This standard allows host systems to get information from the monitor, and to configure the information correctly for the display adapter being used. DDC is expected to become increasingly important in the Macintosh world, and may eventually replace the ADB port and the serial port as the preferred methods of communicating with smart displays. For compatibility with future Macintosh graphics devices, you should follow the standard set by the AppleVision 1710 displays when you are designing displays.

Specifically, the AppleVision 1710 displays implement both DDC1 and DDC2B standards through a scheme that maintains compatibility with the Macintosh sense-code-detection mechanism.

In AppleVision 1710 displays, a 24LC21 serial EEPROM is used to implement DDC, and sense line 1 (pin 10) on the video cable is used for serial data (SDA), while sense line 2 (pin 7) on the video cable is used for the serial clock (SCL).

The displays return the RGB 13” Type 6 (“straight 6”) sense code. (Refer to *Macintosh New Technical Notes HW30 Sense Lines* for more information.) This sense code specifies that sense line 0 is connected to ground, and that sense lines 1 and 2 are unconnected.

These sense lines, therefore, may be conveniently used by DDC, subject to the following restrictions:

- When they are powered up, the displays start in DDC1 mode. The 24LC21 serial EEPROM is ready to clock out one bit of data on sense line 1 (SDA), whenever an edge transition occurs on VSYNC. However, no data is being clocked at this stage, since VSYNC starts after sense determination. This is an important sequence, because if data were being clocked out during sense determination, the Macintosh computer would probably get an incorrect sense reading.
- The computer reads the Type 6 (straight 6) sense of the display. It then goes through a process to determine the level of the Extended Type 6 sense line. As part of this process, the Macintosh pulls sense line 2 low to check if sense lines 1 or 0 are wired to sense line 2. It pulls sense line 1 low to check if sense lines 2 or 0 are wired to sense line 1. At this moment, the 24LC21 EEPROM, detecting that sense line 1 (SDA) is being pulled low externally, automatically switches to DDC2B mode. In this mode, the 24LC21 waits for sense line 2 (SCL) to go low to clock out the first data bit. However, on a graphics device that does not support DDC2B, nothing happens, since sense line 2 has already been pulled low to check for the extended sense code. A graphics device that does support DDC2B is now free to clock out the DDC data.

IMPORTANT

This scheme does not allow “hot-plugging” of video cables. That is, it does not allow you to plug in the video cable after the Macintosh has booted. Be sure to shut down the display before plugging in the video cable or unplugging it. ▲

The video cable pinouts are listed in Table 2-1 on page 21.

Tips for Developers

This section contains miscellaneous information to assist software developers who are supporting smart displays.

IMPORTANT

You do not need to know that a smart display, such as the AppleVision 1710 is attached to your computer. The AppleVision 1710 looks like a 13” RGB display to the video card, and the card does not need to know that it is anything different. If you support invalid timing modes for any sense code, you should add them. The display will adjust the timings to match the timings it supports. ▲

IMPORTANT

You do not need to add new functional sResources. ▲

You should note the following items:

- Support wiggling (tagging). Wiggling is part of the `cscGetConnection` call. It is documented in Chapter 11 of *Designing PCI Cards and Drivers*.

- Do not trim invalid functional sResources. If your video card does not have a programmable ROM, you will need to put the trimmed sResources back in when you patch your driver. These timing modes may be enabled by the display.
- Trim duplicate timings. If your card has different functional sResources with the same timing, you should trim all but one of the functional sResources for each timing mode.
- Trim functional sResources that your hardware cannot support. For example, if your video card does not have enough VRAM to support a functional sResource, trim that sResource rather than marking it invalid. The timing mode represented by that sResource is not one you would want the display to enable.
- Implement `cscGetConnection` and `cscGetModeTiming` for all timing modes. You should mark any timing mode that is not supported by the sense code as invalid and unsafe. When you do this, the display may override the decision.
- Allow the display to switch to invalid timing modes. Some invalid timing modes may be validated by the display, and you should allow the switch, even though you may be unaware of their validity.
- If you do not recognize a sense code, mark all timing modes invalid, and program the hardware with the Apple 13” timing modes. This allows a smart display to come in as an unknown timing mode and enable the modes it supports.

Be particularly alert to the following situations:

- Some drivers use the `kAllModesValid` or `kAllModesSafe` calls from `scsGetConnection`, rather than implementing `cscGetModeTiming` for all timings. If you add a group of invalid modes, you should not mark them as valid in the `cscGetConnection` call.
- If your video card is intended to work on systems that were released before the Display Manager was introduced, you need to check for the Display Manager before enabling invalid timings. Otherwise, the user will see the invalid timings in the Monitors control panel.
- If your video card supports additional standard timings that do not have constants in Video.h, contact Apple Developer Support.

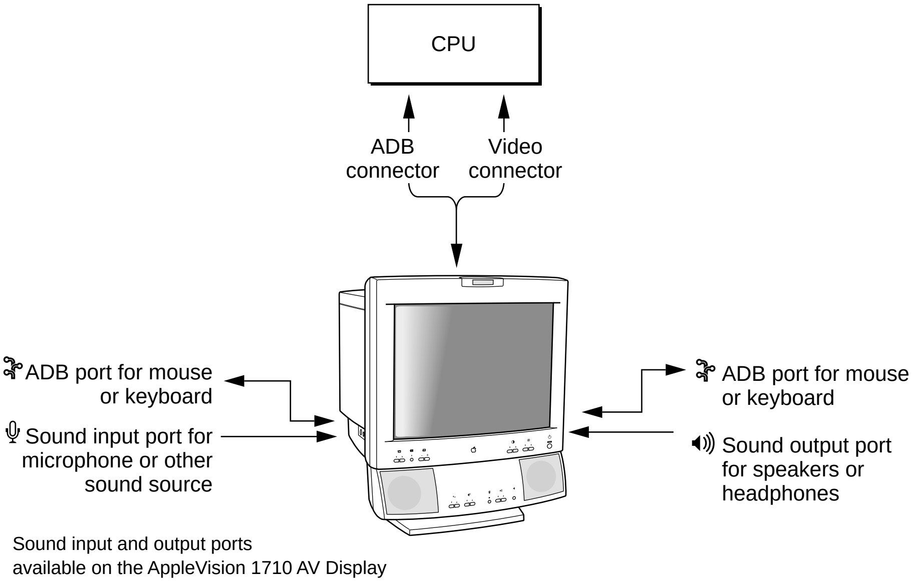
CHAPTER 2

Hardware Interface

This chapter describes the hardware interface for the AppleVision 1710AV and 1710 Displays. Both displays provide an interface for the video connector and the ADB ports and connector. The AppleVision 1710AV Display also provides an interface for the sound input port and the sound output port.

Figure 2-1 gives an overview of the display interface.

Figure 2-1 Display-to-CPU interface



Video Port and Connector

The video port connection is made through a standard DB-15 connector. Figure 2-2 shows the pin designations for the connector, and Table 2-1 lists the signal assignments. The video cable is captive at the display end, and you cannot remove it. The video connector plugs into the video port on the back of the Macintosh computer. If you are connecting the display to a PC-compatible computer, you need a PC adapter that connects to the video cable and then plugs into the video port on the PC. The video cable is part of a dual cable that accommodates both video and ADB features. The ADB features and connector are not used in PC applications.

Figure 2-2 Video connector pin designations

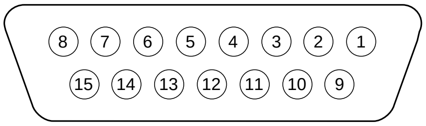


Table 2-1 Video connector signal assignments

Pin number	Output signal
1	Red video ground
2	Red video
3	Not used
4	ID 1 or DDC return
5	Green video
6	Green video ground
7	ID 2 or DDC SCL
8	Not used
9	Blue video
10	ID 3 or DDC SDA
11	VSYNC return
12	VSYNC
13	Blue video ground
14	HSYNC return
15	HSYNC
Shell	Shield ground

Abbreviations:

ID	Identification
DDC	display data channel
SCL	serial clock
SDA	serial data
VSYNC	vertical synchronization
HSYNC	horizontal synchronization

ADB Port and Connector

The ADB cable is part of the dual video and ADB cable. This cable transfers ADB information between the display and the CPU, enabling you to connect ADB devices directly to the display if you wish. The ADB connector is a standard miniature DIN (MD-4) connector. It has four signal pins and an outer shield that functions as chassis ground. Figure 2-3 shows the pin designations for the ADB connector, and Table 2-2 lists the signal assignments.

Note
This ADB cable is used to transfer control data between the Macintosh computer and the display. It is not used in PC applications. ◆

Figure 2-3 ADB connector pin designations

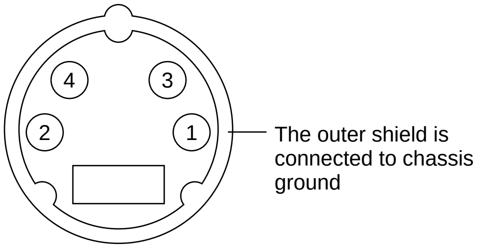


Table 2-2 ADB connector signal assignments

Pin number	Signal name	Description
1	ADB	Bidirectional data bus for input and output; transfers audio (AV display only) and video control data between the CPU and the display
2	POWER ON	Enables power to be turned on from the keyboard
3	+5VDC	+5 V power
4	GND	Logic ground
Outer shield	None	Chassis ground

Other I/O Ports and Connectors

Both the AppleVision 11710AV and 1710 Display have an ADB port on both side panels. These ports allow you to connect a mouse or keyboard directly to the left side of the display

The AppleVision 1710AV display also has the following sound ports:

- a sound input port, on the left panel, allows you to connect an external microphone or other sound source to the display
- a sound output port, on the right panel, allows you to connect speakers or headphones to the display

ADB Ports

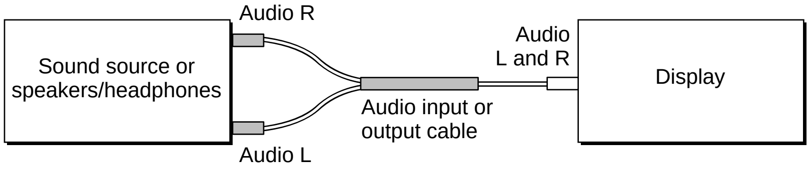
The ADB ports are standard connectors. For ADB port specifications see "ADB Port and Connector" on page 22.

Note
The ADB ports on the side panels of the display are used to connect ADB devices, like the mouse and keyboard. The ADB I/O cable is used to transfer audio and video control data between the Macintosh computer and the display. ◆

Sound Ports and Adapter

The sound ports are found only on the AppleVision 1710AV Display. The sound output port is used for headphones or external speakers. The sound input port accepts sound inputs from a microphone or any appropriate sound source. These ports are both stereo ports, but they use single ministereo audio connectors. You must make sure that the connector you use for audio input or output is compatible with your sound equipment. For example, if you are connecting to a device that uses dual (RCA-type) connectors for stereo sound, use a "Y" adapter to connect the stereo ports of the sound source or speakers to the display's single-connector stereo port. Figure 2-4 shows the sound adapter configuration.

Figure 2-4 Adapter for audio ports



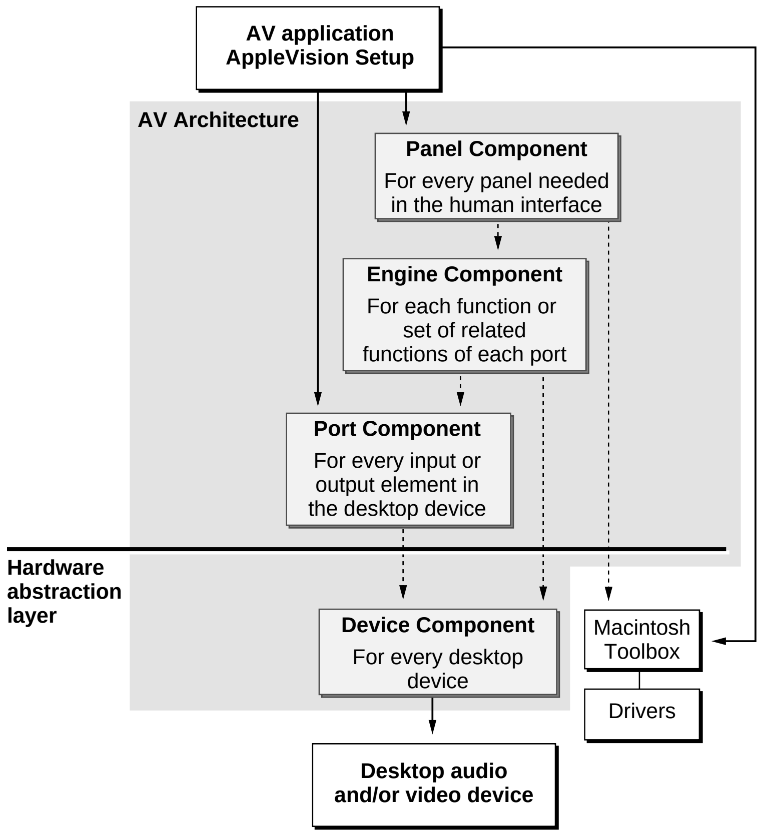
C H A P T E R 3

Application Program Interface

The AppleVision 1710AV and 1710 Displays use a new Macintosh framework known as the AV Architecture. This architecture allows you to access all kinds of audio and video (AV) devices, such as displays, speakers, volume controls, CD players, and so forth. The is described in detail in the *AV Architecture Developer Note*.

The AV Architecture is made up of four basic groups of components: the Panel Components, the Engine Components, the Port Components, and the Device Components. The *AV Architecture Developer Note* describes these components in detail, and explains the relationships between them. Figure 3-1 shows a simplified view of the architecture and its components.

Figure 3-1 Simplified view of the AV Architecture



The AV application used with the AppleVision displays to implement the architecture is known AppleVision Setup. The panels and windows that comprise the applications are based on AV Architecture components, many of which are proprietary. This chapter provides you with information that is useful if you want to modify certain non-proprietary component functions. For information on how to use the AppleVision Setup panels, you should refer to the User’s Guides for the AppleVision displays.

The functions you can modify include:

- Contrast Engine Component functions
- Geometry Engine Component functions
- VPT Engine Component functions

To get a more complete understanding of the AppleVision API and the underlying architectural components, you should read the *AV Architecture Developer Note*. The Component Manager is also critical to your understanding of the information in this chapter, and you can find detailed information about the Component Manager in *Inside Macintosh: More Macintosh Toolbox*, Chapter 6.

ContrastEngine Component

This section describes the component functions contained in the ContrastEngine Component. It is assumed that you are already familiar with the Component Manager and, more specifically, with the AV Architecture to which the ContrastEngine component belongs.

Overview

AppleVision displays have the ability to change contrast and brightness under the control of the Macintosh computer to which they are connected. The ContrastEngine component provides access to the contrast and brightness features of the AppleVision displays, and your application can use this interface to control contrast and brightness settings.

Like other engines belonging to the AV Architecture, the ContrastEngine component is based on Component Manager components. The ContrastEngine component has the following component description:

`componentType = 'avec'`
Indicates that the component is a video panel.

`componentSubType = 'cont'`
Identifies the specific engine.

`componentManufacturer = 'appl'`
Identifies Apple Computer, Inc. as the manufacturer.

ContrastEngine Component Functions

This section describes the ContrastEngine Component functions that control brightness and contrast.

ContrastEngineGetBrightnessRange

This function returns the allowable range of brightness settings for the AppleVision displays. The values returned here can be used to ensure that your application does not attempt to set an invalid brightness value.

```
pascal ComponentResult
ContrastEngineGetBrightnessRange( ComponentInstance
engineComponent, short* min, short* max )
```

`engineComponent`
A valid instance of the ContrastEngine component.

`min, max` Values returned to the caller indicating the range of allowable settings for brightness.

ContrastEngineGetBrightness

This function returns the current brightness setting for the AppleVision displays.

```
pascal ComponentResult
ContrastEngineGetBrightness( ComponentInstance engineComponent,
                             short* brightness )
```

`engineComponent`
A valid instance of the ContrastEngine component.

`brightness` A value returned to the caller indicating the current brightness setting.

ContrastEngineSetBrightness

This function allows your application to set the brightness level for the AppleVision displays.

```
pascal ComponentResult
ContrastEngineSetBrightness( ComponentInstance engineComponent,
                             short brightness )
```

`engineComponent`
A valid instance of the ContrastEngine component.

`brightness` A value passed by your application indicating the desired brightness setting for the display.

ContrastEngineGetContrastRange

This function returns the allowable range of contrast settings for the AppleVision displays. The values returned here can be used to ensure that your application does not attempt to set an invalid contrast value.

```
pascal ComponentResult
ContrastEngineGetContrastRange( ComponentInstance engineComponent,
                                 short* min, short* max )
```

`engineComponent`
A valid instance of the ContrastEngine component.

`min, max` Values returned to the caller indicating the range of allowable settings for contrast.

ContrastEngineGetContrast

This function returns the current contrast setting for the display.

```
pascal ComponentResult
ContrastEngineGetContrast( ComponentInstance engineComponent,
                            short* contrast )
```

`engineComponent`
A valid instance of the ContrastEngine component.

`brightness` A value returned to the caller indicating the current brightness setting.

ContrastEngineSetContrast

This function allows your application to set the contrast level for the display.

```
pascal ComponentResult
ContrastEngineSetContrast( ComponentInstance engineComponent,
                            short contrast )
```

`engineComponent`
A valid instance of the ContrastEngine component.

`brightness` A value passed by your application indicating the desired brightness setting for the display.

GeometryEngine Component

This section describes those GeometryEngine component functions that can be used by the developer. It is assumed that you are already familiar with the Component Manager and, more specifically, with the AV Architecture to which the GeometryEngine component belongs.

Overview

The GeometryEngine component, at present, provides developers with access to the overscan feature of the AppleVision displays. Using this feature, your application can programmatically control the overscan setting.

If you implement the overscan feature on your display, the edges of the image area go as far as they can to the edge of the screen without distorting the image. If you turn on overscan, you cannot make any changes to other Geometry selections, such as height/width adjustment. Normally, you would not use the overscan feature. However, you might want to turn the feature on to view full-screen digital video, or total immersion games. You can also use the feature any time you want to shield the user from details of the computer operation.

Like other engines belonging to the AV Architecture, the GeometryEngine component is based on Component Manager components. The GeometryEngine component has the following component description:

`componentType = 'avec'`
: Indicates that the component is a video panel.

`componentSubType = 'geoe'`
: Identifies the specific engine.

`componentManufacturer = 'appl'`
: Identifies Apple Computer, Inc. as the manufacturer.

GeometryEngine Component Functions

This section describes the GeometryEngine Component functions used to implement overscanning.

GeometryEngineGetOverscan

This function allows your application to determine if overscan is currently on or off.

```
pascal ComponentResult
GeometryEngineGetOverscan ( ComponentInstance engineComponent,
                            short *overscan )
```

`engineComponent`
A valid instance of the GeometryEngine component.

`overscan` A value returned to your application indicating whether the overscan function is currently on or off.

GeometryEngineSetOverscan

This function allows your application to decide whether overscan should be turned on or off.

```
pascal ComponentResult
GeometryEngineSetOverscan ( ComponentInstance engineComponent,
   short overscan )
```

`engineComponent`
A valid instance of the GeometryEngine component.

`overscan` A value passed by your application indicating the desired state of the overscan setting.

VPTEngine Component

This section describes those VPTEngine component functions that can be used by the developer. It is assumed that you are already familiar with the Component Manager and, more specifically, with the AV Architecture to which the VPTEngine component belongs. The section also supplies information on topics associated with the VPTEngine component.

Overview

The VPTEngine component provides access to the color calibration features of the AppleVision displays.

The colors shown on different displays may vary because of minor variances that occur during manufacture. If you are using multiple displays, you will want the colors on all displays to be as closely matched as possible.

AppleVision displays have the ability to correct for manufacturing variances through a process called Virtual Photometry Technology, or VPT. Using VPT, the AppleVision displays can be set to a wide variety of **white points** and can correct for CRT aging, and for **ambient light** conditions. Refer to the following sections, "White Point," "Virtual Photometry Technology (VPT)," and "Ambient Light," for background information on these subjects.

Normally, the VPTEngine component is used by the Accurate Color panel that is in the software provided with AppleVision displays. The information presented in this section is for applications that need specific information about the current color characteristics of the displays. The information returned by thisVPTEngine component interface is unique to the particular AppleVision display being used. It does not supply so-called "nominal" values that may or may not reflect reality.

Like other Engine Components that belong to the AV Architecture, the VPTEngine component is based on the Component Manager. The VPTEngine component has the following component description:

`componentType = 'avec'`
: Indicates that the component is a video panel.

`componentSubType = 'vpte'`
: Identifies the specific engine.

`componentManufacturer = 'aapl'`
: Identifies Apple Computer, Inc. as the manufacturer.

`componentFlags = cmpWantsRegisterMessage`
: Makes sure the component is suitable for the machine.

White Point

The image on your display is a combination of red, green, and blue signals. All lights, including display light, have a white point, which is the measure of the color content of the light. The AppleVision displays allow you to change the intensity (or white point) of the R (red), G (green), and B (blue) signals. White point is measured in degrees Kelvin, and it sets the foundation for the other colors on your display. If you have a high white point, colors have a bluish tinge. If you have a low white point, colors have a slightly reddish tinge.

Virtual Photometry Technology (VPT)

VPT is a proprietary Apple technique used to calculate a theoretical white point. It does this by measuring the currents for the R, G, and B electron guns, while displaying a white screen. The current values are correlated to factory calibration data that is programmed into the display during production. An extremely accurate white point can be determined by compensating for the aging affects of the display. The current sensing circuitry also allows the host CPU to calculate the observed color of any pixel on the screen.

Ambient Light

Ambient light is the light surrounding your display, and it makes a difference to the way colors appear on the screen. Ambient light may be normal window light, sunlight, incandescent light, fluorescent light, and so on. To get a true color rendering, you should first set the white point for the display and then correct for ambient light.

VPTEngine Component Data Types

This section describes the data structures unique to the VPTEngine component. They are `Tristim (XYZ)`, `CIE1976 (u'v')`, `CIE1960 (uv)`, `CIE1931 (xy)`, `Kelvin`, and `VPTGamma`.

These data structures define the "color space " of the display. Over the years, a number of notations for colors have been developed. Most of these notations are considered to be color spaces because they define absolute three-dimensional spaces where colors reside. The standard RGB notation used on the Macintosh (and on many other computers) is not a real color space because it has no absolute representation. In other words, if R, G, and B are all set to maximum, you usually get white. However, these settings do not define exactly the shade of white you get, since there are many definitions of white.

The Commission International de l'Eclairage, also known as the International Commission on Illumination, and referred to in this developer note as the CIE Committee, has created some of the most useful color standards. These are the color space standards referenced in the following sections.

Tristim (XYZ)

This data structure is used to represent the Tristimulus color space. This color space was first defined in 1931 by the CIE committee.

```
typedef struct Tristim
{
   doubleX;
   doubleY;
   doubleZ;
} Tristim;
```

`X, Y, Z`	Standard elements of the Tristimulus (XYZ) notation.

CIE1976 (u’v’)

This data structure is used to represent the color space defined by the CIE committee in 1976.

```
typedef struct CIE1976
{
   doubleuPrime;
   doublevPrime;
   doublefL;
} CIE1976;
```

`uPrime, vPrime`	Coordinates in the standard CIE 1976 (u’v’) notation.
`fL`	Foot Lamberts, a standard unit that describes luminance.

CIE1960 (uv)

This data structure is used to represent the color space defined by the CIE committee in 1960.

```
typedef struct CIE1960
{
   doubleu;
   doublev;
   doublefL;
} CIE1960;
```

`u, v`	Coordinates in the standard CIE 1960 (uv) notation.
`fL`	Foot Lamberts, a standard unit that describes luminance.

CIE1931(xy)

This data structure is used to represent the color space defined by the CIE committee in 1931. Do not confuse it with the Tristimulus format, also defined by the CIE in 1931.

```
typedef struct CIE1931
{
   doublex;
   doubley;
   doublefL;
} CIE1931;
```

`x, y`	Coordinates in the standard CIE 1931 (xy) notation.
`fL`	Foot Lamberts, a standard unit that describes luminance.

Kelvin

This data type represents Kelvin degrees, a simple unit of measuring a very narrow range of colors.

```
typedef long Kelvin;
```

VPTGamma

This data structure is used by the VPTEngine component to represent a gamma table. A **gamma curve** defines the relationship between the color intensity (**chrominance**) of the screen and the light intensity (**luminance**) of the screen. With a low gamma curve, colors may look washed out to the user, while a high gamma curve provides more contrast. The human eye can determine subtle changes in color, but does not register these changes in a linear way. Color systems are linear and continuous. A linear gamma curve ensures that the display produces the correct luminance levels on each primary color channel, and matches the user’s nonlinear color expectations to the display’s linear color devices.

```
typedef struct VPTGamma
{
   unsigned shortred[256];
   unsigned shortgreen[256];
   unsigned shortblue[256];
} VPTGamma, *VPTGammaPtr, **VPTGammaHdl;
```

`red, green, blue`
: Arrays containing the gamma curve for each electron gun of the display.

VPTEngine Component Functions

This section describes the VPTEngine components functions associated with

- calibration
- white-point settings
- ambient-light settings
- gamma-curve settings
- color gamut identification
- translating standard RGB QuickDraw colors to Tristimulus color values, and vice versa

VPTEngineCalibrateDisplay

This function allows the caller to force the display to go through a calibration cycle. Your application probably will not need to call this function, since the software included with the AppleVision display causes the display to be recalibrated automatically every two weeks or so.

```
pascal VPTEngineResult
VPTEngineCalibrateDisplay( VPTEngineComponent engineComponent
```

`engineComponent`
A valid instance of the VPTEngine component.

Supplementary Information

Calling this function causes the entire Macintosh system to stop whatever it is doing and spend several moments recalibrating the display. During this time, the entire screen flashes black, white, and several other colors.

VPTEngineCalibrated

This function allows your application to determine whether or not the display is currently in a calibrated state.

```
pascal VPTEngineResult
VPTEngineCalibrated( VPTEngineComponent engineComponent,
      Boolean* calibrated )
```

`engineComponent`
A valid instance of the VPTEngine component.

`calibrated` Set to true if the display is in a calibrated state.

Supplementary Information

In general, the only time this function will return `false` for the `calibrated` parameter is when the AppleVision software finds itself with no preference file on the startup drive and has not yet asked the user to recalibrate. This brief interval usually lasts about 15 minutes (the amount of time it takes the display to warm up fully) but for color-critical applications, this function should be called before relying on the information provided through the VPTEngine component interface.

VPTEngineGetLastCalTime

This function returns the date & time (represented as the number of seconds elapsed since midnight, January 1, 1994) that the display was last calibrated.

```
pascal VPTEngineResult
VPTEngineGetLastCalTime( VPTEngineComponent engineComponent
            unsigned long* lastCalTime )
```

`engineComponent`
A valid instance of the VPTEngine component.

`lastCalTime` The date and time of the most recent calibration.

VPTEngineGetTargetWhitePoint

This function allows your application to determine the user-set white point for the display. The user-set white point differs from the actual white point achieved by the display's calibration process.

```
pascal VPTEngineResult
VPTEngineGetTargetWhitePoint( VPTEngineComponent engineComponent,
                    CIE1931* whitePoint )
```

`engineComponent`
A valid instance of the VPTEngine component.

`whitePoint` A structure that is passed back to the caller containing the target white point for the display.

Supplementary Information

For most uses, the target white point and the actual white point should be fairly interchangeable, although it is better to use the actual white point for maximum color accuracy.

VPTEngineGetActualWhitePoint

This function allows your application to determine the actual white point achieved by the display's calibration process. This differs from the target white point, described above, and includes the effects of **ambient light** on the display's image surface.

```
pascal VPTEngineResult
VPTEngineGetActualWhitePoint( VPTEngineComponent engineComponent,
                    CIE1931* whitePoint )
```

`engineComponent`
A valid instance of the VPTEngine component.

`whitePoint` A structure that is passed back to the caller containing the actual measured white point for the display.

Supplementary Information

For most uses, the target white point and the actual white point should be fairly interchangeable, although it is better to use the actual white point for maximum color accuracy. In the event that the AppleVision display is unable to match the white point requested by the user perfectly, the white point returned by this function will closely reflect the white point being emitted by the display.

VPTEngineGetWhitePointDrive

This function allows your application to determine the white point drive currently being emitted from the display. This differs from the actual white point in that the white point drive does not include the effects of ambient light, even if the user has indicated that ambient light is present.

```
pascal VPTEngineResult
VPTEngineGetWhitePointDrive( VPTEngineComponent engineComponent,
                  CIE1931* whitePointDrive )
```

`engineComponent`
: A valid instance of the VPTEngine component.

`whitePointDrive`
: A structure that is passed back to the caller containing the measured white point drive for the display.

Supplementary Information

White point drive is the color of white that would be emitted by the display if it were placed in a dark room with no ambient light. Since the effects of ambient light can cause significant shifts in the colors viewed by the user, your application should probably use the `VPTEngineGetActualWhitePoint` function, which takes into account the effects of ambient light.

VPTEngineGetReflectedAmbientLight

This function allows your application to determine the amount of ambient light being reflected back at the user from the face of the display. The user determines the amount of light being reflected, by means of the Accurate Color control panel.

```
pascal VPTEngineResult
VPTEngineGetReflectedAmbientLight( VPTEngineComponent
engineComponent, CIE1931* reflectedAmbientLight )
```

`engineComponent`
: A valid instance of the VPTEngine component.

`reflectedAmbientLight`
: A structure that is passed back to the caller containing the color and intensity of the ambient light being reflected back at the user from the face of the display.

▲ **WARNING**
Since the information returned by this function is essentially determined by the user (with the help of the Accurate Color control panel), your application should be cautious about assuming the information is absolutely accurate. ▲

VPTEngineGetDirectAmbientLight

This function allows your application to determine the amount of ambient light falling on the surface of the display. The user determines the amount of light using the Accurate Color control panel.

```
pascal VPTEngineResult
VPTEngineGetDirectAmbientLight( VPTEngineComponent
engineComponent,
            CIE1931* directAmbientLight )
```

`engineComponent`
A valid instance of the VPTEngine component.

`directAmbientLight`
A structure that is passed back to the caller containing the color and intensity of the ambient light falling on the surface of the display.

▲ **WARNING**
Since the information returned by this function is essentially determined by the user (with the help of the Accurate Color control panel), your application should be cautious about assuming the information is absolutely accurate. ▲

VPTEngineGetActualGammaCurveTable

This function gives your application access to a table representing the measured gamma curve of this monitor.

```
pascal VPTEngineResult
VPTEngineGetActualGammaCurveTable( VPTEngineComponent
engineComponent,
            VPTGammaPtr actualGamma )
```

`engineComponent`
A valid instance of the VPTEngine component.

`actualGamma` A structure that is passed back to the caller containing the color and intensity of the ambient light falling on the surface of the display.

Supplementary Information

The gamma curve table returned by this function is the system gamma curve table. In other words, the data returned represents the overall gamma curve produced by the combination of the particular display and the video output device to which it is attached. This differs from the **gamma correction**, which is not available from the VPTEngine component. Gamma correction is the technique that adjusts the gamma curve to compensate for the loss of detail in dark objects.

VPTEngineGetActualGamut

This function allows your application to determine the **gamut** the display is capable of reproducing.

```
pascal VPTEngineResult
VPTEngineGetActualGamut( VPTEngineComponent engineComponent,
            Tristim* red,
            Tristim* green,
            Tristim* blue )
```

`engineComponent`
: A valid instance of the VPTEngine component.

`red`
: A structure that is passed back to the caller containing the color and intensity of the brightest red the display is capable of reproducing.

`green`
: A structure that is passed back to the caller containing the color and intensity of the brightest green the display is capable of reproducing.

`blue`
: A structure that is passed back to the caller containing the color and intensity of the brightest blue the display is capable of reproducing.

Supplementary Information

The gamut is the entire range of colors that can be represented by a given device. For displays, the gamut can be defined as the area enclosed by three points in a color space, since there are three electron guns (R, G, and B) in the display.

VPTEngineRGBToTristim

This function allows your application to translate a standard RGB QuickDraw color into an equivalent Tristimulus color value for the display. In effect, it allows your application to translate an RGB QuickDraw color into a real color space.

```
pascal VPTEngineResult
VPTEngineRGBToTristim( VPTEngineComponent engineComponent,
               RGBColor* theColor,
               Tristim* theTristim )
```

`engineComponent`
: A valid instance of the VPTEngine component.

`theColor`
: A structure passed to this function containing an RGB QuickDraw color.

`theTristim`
: A structure that is passed back to the caller containing the equivalent Tristimulus color value.

Supplementary Information

RGB QuickDraw colors are essentially arbitrary. It is somewhat difficult to say what actual color will be produced when a particular RGB QuickDraw color value is displayed.

VPTEngineTristimToRGB

This function allows your application to translate a Tristimulus color value into the equivalent RGB QuickDraw value for the display.

```
pascal VPTEngineResult
VPTEngineTristimToRGB( VPTEngineComponent engineComponent,
      Tristim* theTristim,
      RGBColor* theColor )
```

`engineComponent`
: A valid instance of the VPTEngine component.

`theTristim`
: A structure passed to this function containing the coordinates of a color represented in the Tristimulus color space.

`theColor`
: A structure that is passed back to the user containing the equivalent RGB QuickDraw Color.

Glossary

ambient light This is the light surrounding your display. It may be normal window light, sunlight, incandescent light, fluorescent light, and so on. The ambient light makes a difference in the way colors appear on the screen.

chrominance This component of the picture information contains only the color and no picture detail.

CIE (Commission International de l'Eclairage) CIE is the International Commission on Illumination. The standards defined by the CIE are specified in terms of the year they were created. For example, the 1931 CIE color model is the standard created in 1931.

ColorSync This is a system extension that provides color-conversion capabilities and improves color consistency. ColorSync translates the colors used on one device, such as the AppleVision 1710AV Display, so that they match the colors displayed on another display, or printed on a color printer.

ColorSync profile This is the profile for the display that is automatically installed as part of the display software. You can create a ColorSync profile that describes the white point setting for each individual display. You can use this profile with any display that supports Apple's ColorSync color matching system.

Energy Saver Energy Saver 1.1 is an energy conservation control panel that allows you to tell the display to conserve energy after a specified period of inactivity. If you turn on Energy Saver, the screen dims when the keyboard or mouse has been inactive for a specified period of time. You can access the Energy Saver panel from Control Panels in the Apple () menu.

Energy Star When the display detects the loss of certain signals from the CPU, it goes into a low-energy sleep mode. The U.S. Environmental Protection Agency recently initiated a certification program called Energy Star to recognize products that support this power-saving mode. The EPA Energy Star logo on the display indicates that it complies with the EPA's Energy Star program.

fixed-frequency displays Also known as single-mode displays, these displays are capable of supporting only one frequency or resolution. The Apple RGB 13" display is an example of this type of display. See also **multiple-scan displays**.

gamma correction This is a technique that adjusts the **gamma curve** to compensate for the loss of detail in dark objects.

gamma curve This is the relationship between color intensity (**chrominance**) and light (**luminance**). With a low gamma curve, colors are washed out. With a high gamma curve, colors have more contrast.

gamut The full range of colors the display can produce on the screen. Sometimes referred to as the color gamut.

Kelvin A temperature scale used in scientific applications. Used to measure white point in the color display.

luminance This component of the picture information is responsible for detail, shapes, and shadings.

MovieTalk Apple's trademark for a network protocol designed to handle audio and video data streams efficiently. It is part of QuickTime Conferencing. The MovieTalk protocol establishes a set of rules for setting up, maintaining, and breaking down a connection and for delivering media data. It consists of a media stream protocol and a connection control channel protocol. The MovieTalk protocol is platform and media independent.

multiple-scan displays These displays can support a range of frequencies and resolutions, for example the AppleVision 1710AV and 1710 Displays. See also **fixed-frequency displays**.

over scan Overscan moves the screen image as close to the edges of the screen as possible without distorting the image. This setting is useful when you are using the AppleVision displays to view videos.

pincushioning This term describes distortions of the screen image, where the sides of the image either bulge out to create convex pincushioning, or are pulled in to create concave pincushioning.

smart displays These displays have features, such as brightness and contrast, that can be controlled from the host computer. The AudioVision 14 display was Apple's first smart display.

Virtual Photometry Technology (VPT) A technique used to calculate a theoretical white point by measuring the currents for the R (red), G (green), and B (blue) electron guns while displaying a white screen. The current values are correlated to factory calibration data programmed into the monitor during production. An extremely accurate white point can be determined by compensating for the aging affects of the display. The current sensing circuitry also allows the host CPU to calculate the observed color of any pixel on the screen.

white point The image on your display is a combination of red, green, and blue signals. All lights, including display light, have a white point, which is the measure of the color content of the light. The AppleVision 1710 displays allow you to change the intensity, or white point of the R, G, and B signals. White point is measured in degrees Kelvin, and it sets the foundation for the other colors on your display. At a high white point, colors have a slightly bluish tinge, while at a low white point, they have a slightly reddish tinge.

wiggling Refers to a process of changing, or toggling, the level of the display's sense line, to alert the computer to the fact that the display is present. The level of the sense line is changed from 0 to 1 or from 1 to 0, and then back to its original level. In some Apple publications, the term tagging is used instead of wiggling.

Index

A

B

C

This Apple manual was written, edited, and composed on a desktop publishing system using Apple Macintosh computers and FrameMaker software. Proof pages and final pages were created on the Apple LaserWriter Select. Line art was created using Adobe™ Illustrator. PostScript™, the page-description language for the LaserWriter, was developed by Adobe Systems Incorporated.

Text type is Palatino® and display type is Helvetica®. Bullets are ITC Zapf Dingbats®. Some elements, such as program listings, are set in Apple Courier.

WRITER
Joyce D. Mann

DEVELOPMENTAL EDITORS
John Hammett

ILLUSTRATOR
Sandee Karr

Thanks to
Ross Lemcke, Jesse Devine, Ian Hendry, Ingrid VanCleemput, and Lisa Kelly,
Special thanks to
Mark Taylor